AMENDMENTS TO THE CLAIMS

1. (Previously Presented) A multi-threading processor, comprising:

a first instruction fetch unit to receive a first thread and a second instruction fetch unit to receive a second thread;

an execution unit to execute said first thread and said second thread; and

a multi-thread scheduler coupled to said first instruction fetch unit, said second instruction fetch unit, and said execution unit, wherein said multi-thread scheduler is to determine the width of said execution unit.

2. (Original) A multi-threading processor as recited in claim 1, wherein the multi-thread scheduler unit determines whether the execution unit is to execute the first thread and the second thread in parallel depending on the width of the execution unit.

3. (Original) A multi-threading processor as recited in claim 2, wherein the multi-thread processor is an in-order processor.

4. (Original) A multi-threading processor as recited in claim 3, wherein the execution unit executes the first thread and the second thread in parallel.

5. (Original) A multi-threading processor as recited in claim 3, wherein the execution unit executes the first thread and the second thread in series.

6. (Original) A multi-threading processor as recited in claim 3, wherein the first thread and the second thread are compiled to have instruction level parallelism.

7. (Original) A multi-threading processor as recited in claim 6, further comprising:

a first instruction decode unit coupled between the first instruction fetch unit and the multi-thread scheduler; and

a second instruction decode unit coupled between the second instruction fetch unit and the multi-thread scheduler.

8. (Original) A multi-threading processor as recited in claim 4, wherein the execution unit executes only two threads in parallel.

9. (Previously Presented) A computer implemented method, comprising:

determining whether a multi-threading processor is wide enough to execute a first thread and a second thread in parallel; and

executing said first thread and said second thread in parallel if said multi-threading processor is wide enough to execute the first thread and the second thread in parallel.

10. (Previously Presented) The method as recited in claim 9, further comprising executing the first thread and the second thread in series if said multi-threading processor is not wide enough.

11. (Previously Presented) The method as recited in claim 10, wherein the multi-threading processor is an in-order processor.

12. (Previously Presented) The method as recited in claim 11, further comprising compiling the first thread and the second thread, wherein the first thread and the second thread have instruction level parallelism.

13. (Previously Presented) The method as recited in claim 12, wherein the multi-threading processor executes only two threads in parallel.

14. (Previously Presented) The method as recited in claim 13, further comprising:

fetching the first thread and the second thread; and

decoding the first thread and the second thread.

15. (Previously Presented) A set of instructions residing in a storage medium, said set of instructions to be executed by a multi-threading processor for searching data stored in a mass storage device comprising:

determining whether said multi-threading processor is wide enough to execute a first thread and a second thread in parallel; and

executing said first thread and said second thread in parallel if said multi-threading processor is wide enough to execute the first thread and the second thread in parallel.

16. (Previously Presented) A set of instructions as recited in claim 15, further comprising executing the first thread and the second thread in series if said multi-threading processor is not wide enough.

17. (Previously Presented) A set of instructions as recited in claim 16, wherein the multi-threading processor is an in-order processor.

18. (Previously Presented) A set of instructions as recited in claim 17, further comprising compiling the first thread and the second thread, wherein the first thread and the second thread have instruction level parallelism.

19. (Previously Presented) A set of instructions as recited in claim 18, wherein the multi-threading processor executes only two threads in parallel.

20. (Previously Presented) A set of instructions as recited in claim 19, further comprising:

fetching the first thread and the second thread; and

decoding the first thread and the second thread.

21. (Previously Presented) A system comprising:

a memory to store a set of instructions; and

a processor coupled to the memory to execute the set of instructions, the processor with a first instruction fetch unit to receive a first thread, a second instruction fetch unit to receive a second thread, an execution unit to execute said first thread and said second thread, and a multi-thread scheduler coupled to said first instruction fetch unit, said second instruction fetch unit, and said execution unit, wherein said multi-thread scheduler is to determine the width of said execution unit.

22. (Previously Presented) The system of claim 21, wherein the multi-thread scheduler unit determines whether the execution unit is to execute the first thread and the second thread in parallel depending on the width of the execution unit.

23. (Previously Presented) The system of claim 22, wherein the multi-thread processor is an in-order processor.

24. (Previously Presented) The system of claim 23, wherein the execution unit executes the first thread and the second thread in parallel.

25. (Previously Presented) The system of claim 23, wherein the execution unit executes the first thread and the second thread in series.

26. (Previously Presented) The system of claim 23, wherein the first thread and the second thread are compiled to have instruction level parallelism.

27. (Previously Presented) The system of claim 26, further comprising:

a first instruction decode unit coupled between the first instruction fetch unit and the multi-thread scheduler; and

a second instruction decode unit coupled between the second instruction fetch unit and the multi-thread scheduler.

28. (Currently Amended) ~~A~~ The system of claim 24, wherein the execution unit executes only two threads in parallel.